

THE SURVEY - PART 3

by Lynn & Buzz Rudow

This survey is really getting interesting. Many thanks to those of you who responded. We now have 43 of the 149 members responding, for a 29% ratio.

If you members who haven't sent in your questionnaire care to submit it now, we'll include your responses in the continuing, and the final, analysis. Although we start coming up with conclusions after we document the responses to each question, we'll refrain from analysis until we get to the end of the data presentation.

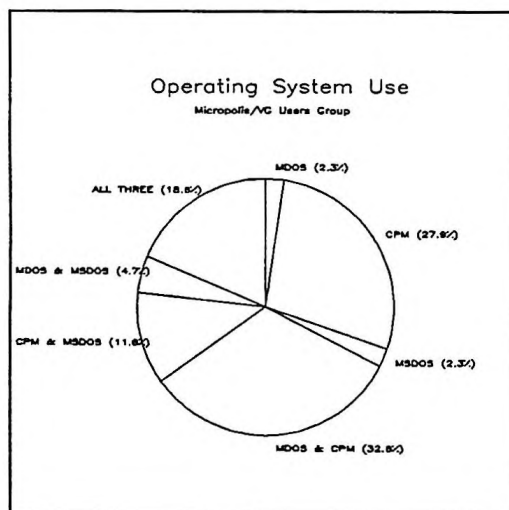
We'll now continue passing you the information by taking up where we left off, that being at question #2.

2. What operating system do you use?

The pure numbers are as follows:

MDOS only	02.3%
CP/M only	27.9%
MS-DOS only	02.3%
MDOS & CP/M	32.6%
CP/M & MS-DOS	11.6%
MDOS & MS-DOS	04.7%
MDOS,CP/M, MS-DOS	18.6%

This distribution is shown graphically in the following pie chart.



It's a bit surprising to us that we have so many MDOS users. MDOS, for those of you who don't go back that far, is the Micropolis Disk Operating System. It is tied to the 16-hard-sector controller. As far as we know, it's never been installed on a system with the

Tandon drives, though we don't know any particular reason why it hasn't.

The question read, "what system do you *use*", not own, so we assume that means the above percentage of people are using MDOS to some extent. Those users haven't been making much noise lately. We thought everyone had abandoned MDOS.

Another interesting statistic is shown when you add up the percentages that use only one system, and those that use more than one system.

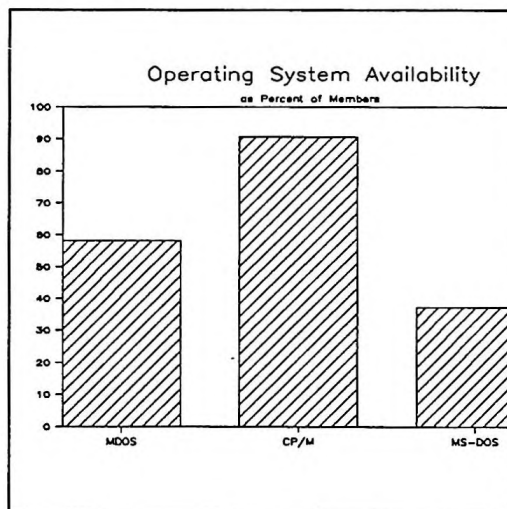
Only one	32.5%
More than one	67.5%

It's also obvious that a lot of people are using MS-DOS, even though it isn't a topic of this publication.

The percentage of users having availability of each of the three operating systems is:

MDOS	58%
CP/M	91%
MS-DOS	37%

or, graphically:



There are a few liberties we've taken in presenting this information. We combined references to PC-DOS with MS-DOS, and if both systems belonged to a single member, we only counted once.

Perhaps a more flagrant liberty is the combining of references to XCP/M, CP/M-86, and CCP/M-86 with CP/M. XCP/M, being Extended CP/M, or Vector's version of CP/M-3, still runs on an 8-bit machine. CP/M-86 and Concurrent CP/M-86 are 16-bit operating systems, however.

Our logic was that these systems were still being run on Vector equipment which can also run 8-bit CP/M programs as a task. We've assumed that all references to MS-DOS implied an IBM-type equipment change, which doesn't run 8-bit CP/M. Well, it can, but it takes additional implementation of hardware and/or software.

If you're keeping score, 4.7% of the members are running CP/M-86. In every case those members also had 8-bit CP/M and MS-DOS.

3. *Do you program, have someone else program for you, or buy packaged software?*

One would expect 88.3% of the respondents to answer this question, since that's the percentage that said they wrote software in question 3. Actually, 90.7% responded. We assume the extra 2% of the respondents program in a learning mode and don't write their application programs.

The pure numbers are as follows:

Write own	37.2%
Buy	11.6%
Write & Buy	51.2%

An interesting item here is that only one member responded that he had someone else program for him. Our experience is that packaged software won't always efficiently do what we want it to do. It is too generalized, or horizontal (across the board, all types of users), in nature. What we need is specialized, or vertical (applies to a specific task or type of business) software. You can purchase vertical software. But, for smaller tasks, a personally-created package can be better than even the commercial vertical software. The personally-created software can be smaller in size than the commercial package, faster, and provide more precise data organization and product design.

Evidently, everyone (88.3% of you) is capable of programming well enough to do this production for themselves.

4. *What languages do you program in?* (most program in more than one)

BASIC	98%
PASCAL	18%
ASM	50%
C	16%
FORTRAN	18%
COBOL	08%

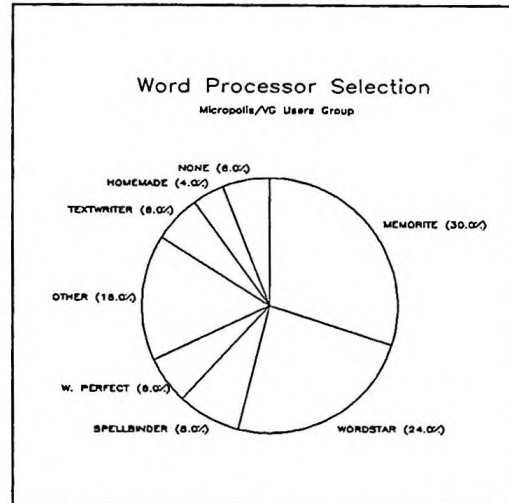
Forth, Machine and C-Pistol were others.

5. *What wordprocessor do you use?* (a lot of you use more than one)

Memorite	57%
Wordstar	41%

Word perfect	08%
Spellbinder	08%
Textwriter	05%
Scope	05%

Word Pal, Word Master, Speed Script, Nevada Edit and Homebrew were the others used.



That's all for now. We'll continue next month.

□ □ □

LIBRARY DISK SUBMITTALS

by Ed King, PO Box 787,
Ithaca, N.Y. 14851 (607) 273-5577

I've submitted a bunch of programs and files to Buzz for the Micropolis/Vector Graphic Public Domain Disk Library. What follows is a brief description of some of the contents.

INDEXING MEMORITE FILES

MEMINDX1.COM and MEMDEX.COM are two of the earlier versions of the assembly language programs which I have written to gather from a disk the filenames and descriptors (if any) of all [or selected] .MEMORITE text files. The program finds, collects, alphabetizes by filename, and prints the .MEM filename and descriptive data to the screen -- and, if elected, also into a diskfile. Of course a CTRL P will send the screen output to the printer as well.

Even if you don't use the word processor MEMORITE.COM, you can try these programs out on this disk, for I have put a number of short .MEM-type files on the disk, and I have created a demo file of the collection, called DEMO/MEM.MEM -- which you may

also use in experimenting with FIND! program, described below.

Wildcards are permitted in specifying the filenames to be collected by MEMINDEX1 and MEMDEX. Try the following commands directed to the files on this disk:

```
A>B:MEMDEX EU*
B>MEMINDEX1 PL*
B>MEMINDEX1 MEM*
```

As you can see, you can build (on your default disk) files of selected filename descriptors: so if you have some system in naming your Memorite text files, you will be able to collect selective disk files of their names and descriptors.

The program does not [yet] permit you to direct the output file to a particular drive: it writes its output file(s) to the default drive. Therefore, to obtain a MEMINDEX file of the .MEM files on each of several disks, it is most convenient to put the program disk in the A: drive, select B: as the default drive, and run the disks to be so "catalogued" through the B: drive (having issued a ^C warmboot after the disk swap, of course, to enable a disk write). The form of the command:

```
B>A:MEMINDEX [drive:][filename]
```

Omit any reference to filetype, as the program assumes only a .MEM filetype. As stated, [filename] may include wildcards ? or *.

The descriptive data picked up by MEMINDEX1 from .MEM files is part of that listed to screen by the directory function in MEMORITE. The version of the program called MEMDEX produces a shorter such "index", omitting dates, etc.

The resulting output file is given a .MEM filetype so that it can easily be brought up in Memorite and edited, if desired -- though if it is desired to incorporate its data in a data base, the length of each field must be observed and preserved, thus requiring that you keep careful count while you edit, and that you not slip in an extra space somewhere. It is easier to let the MEMORITE program do the counting and spacing by using the MEMORITE ID command -- even though to change the descriptor with it, you have to give the file a new filename, at least temporarily, and have enough room left on your disk to record the file again under the new name, unless you are willing to first erase the old filename. Though a bit cumbersome, this procedure will insure against an unintended change in the length of an entry.

The public domain program FIND! may be used on the MEMDEX and MEMINDEX output files in order to pick out the entries (strings from filenames, descriptors, dates) sought. (See the "DEMONSTRATE" suggestions in the documentation on FIND!.) If you have been consistent in describing your text files -- per-

haps even incorporating keywords in the descriptors -- or if you have a good memory or instinct for the a word or part of a word you might have used in a MEMORITE file descriptor, or you know the date of creation or last edit of a file, these MEMINDEX disk files and the program FIND! will provide you with a great electronic Index.

You might wish to improve on your file descriptors, as suggested above, before incorporating them into disk files with one of the MEMINDEX programs. Though you will have had to give the file a new name in order to give it a new description, once back in CP/M you can easily RENAME your file to its original name, if you wish.

A BUG exists in these versions of MEMINDEX1 and MEMDEX. They will crash if any .MEM file to be picked up is a 0K (empty) file. To avoid (or recover), use a Directory program like D.COM, SD.COM, or STAT.COM to check the size of all .MEM files on the target disk, erasing any which are 0K files before running the MEMINDEX program.

Included on the library disk will be a number of short .MEM files to give you a DEMO of the programs even if you do not use MEMORITE. Have fun!

SORTED DIRECTORY PROGRAM

The SD.COM program on this library disk is the public domain Sorted and Sized directory program, of course. This particular implementation does not do an automatic disk reset: so you must, if you want accuracy in the disk usage figures it returns, have either reset the drive, OR have given a drive designator on the command line, followed by a space and \$R option. Thus:

```
A>SD B:[filename.filetype] $R
```

You may use wildcards for the filename or the filetype.

EXTENDING YOUR CP/M

ICEX is a kind of forerunner to ZCPR3, independently invented and differing from it, I gather. It is Public Domain. I read of it in the first re-start issue of MICRO-SYSTEMS back in early 1985, where the source code was published: but its author, Dave Brewer, sent it to me on an IBM standard disk, with an EXTENSIVE MANUAL, for \$18.00. I have yet to view and use it. There is a DOC file on here to clue you into its multitude of parameters, options and ver-satilities.

COUNTING BITS

The file CRCKLIST.TXT file contains the CRC of most of the files on this disk: it was created by B>CRCK.COM F and renamed from the CRCKLIST.CRC filename which CRCK's "F" option produces. Please pardon me if I am covering ground that is probably 'Old Hat' to you, or is of no interest.

Better than no explanation where one would be appreciated.

FINDING TEXT STRINGS

FIND!.COM is a slight modification of the Ward Christenson string search program to make it accept multiple strings on the command line on a Vector Graphic 3 system. This version calls for the use of an Exclamation Point [!] as the separator of [viz. a Logical OR between] the strings being searched for -- because the Vector 3 does not seem to be able to generate (on the command line, where you need it) the Vertical Bar (or PIPE) (=ASCII 7CH) which the FIND.COM uses as the normal separator (= "OR" sign). [The pipe is generated by CTRL U in Memorate; but on the CP/M 2.22 H command line, a ^U(15H) generates a # sign and CR LF -- giving you not a PIPE, but a new command line.]

See the FIND.DOC file for the uses of FIND!. I was impelled to modify FIND!.COM in order to search Disk Files containing the .MEM-file descriptors and filenames -- such disk files having been created by my new programs, MEMDEX.COM and MEM-INDEX.COM (or "MEMINDXn.COM" -- the developmental versions), which are on this disk. FIND! will pick up any specified string(s), and so can locate filename and descriptive words or strings with a mere scrap to go on.

To DEMONSTRATE the use of FIND!, I have created on this disk (using MEMINDX1.COM) a file (DEMO/MEM.MEM) of all the .MEM files on the disk, and their descriptors. Try the following commands for a demonstration:

```
B>FIND! [drive:]DEMO/MEM.MEM z!248
B>find! [drive:]DEMO/MEM.MEM z !248
B>find! [drive:]demo/mem.mem me !04!127
B>find! demo/mem.mem me!04!127
B>find! demo/mem.mem NONE!LIFE!LOVE
B>FIND! DEMO/MEM.MEM 04/09/86
```

A single string for the search may be designated, of course; but do not end it with the exclamation point, for it will interpret that as designating a search for all SPACES!

RESCUING CRASHED-PROGRAM TEXT

The SAVxxx.COM programs and their source code in the .ASM files are adaptations of the David Weinberger "SAVESTAR" program which he wrote and published for private, non-profit use in PROFILES, February 1986, pg 48, as part of his article "Getting Started with Assembly Language". He named it SAVESTAR, having set it to save WORDSTAR text; but he provided in the .ASM file the values needed to reset and assemble it to rescue text files created by a number of other programs, and he invited all to have a go at making improvements in the program. He included the equates (EQU values)

for several versions of TURBO PASCAL; I added those for SCOPE ver 1.9, MEMORITE vers. 1.2 and 1.4; and CONECT.

The program is designed to rescue, by writing it to a diskfile named SALVAGED, whatever text existed in consecutive memory when the generating program crashed. It can be tested merely by doing a reset after you have generated some text in memory with a particular program for which the SAVExxx.COM program has been designed, and then doing a warmboot from MON> (viz. via an "L" command for MON 4.2) to get back to CP/M to run the SAVExx program.

Weinberger's published version is a mini-version of his larger program. He invites you to copy, modify, and enhance the program as an assembly language learning exercise; and as a tutorial, he takes you through a detailed explanation of the function of each section of his code.

I have changed the program to provide user control over whether an existing "SALVAGED" file should be erased for the new one.

I also changed the format of the sign-on message, and rather than using a static BDOS page address as the ultimate MAXimum or ceiling point for buffer capture, I have fixed the program so it picks up that current BDOS base address from memory location 0007h -- so that if you have some program residing below CP/M (e.g. SMARTKEY or a RAM-disk driver) the SAVExx program will not erroneously include that program code in the data it saves -- even though it would not be a catastrophe for it to do so, for such 'garbage' could later be erased from the SALVAGED file.

If one can locate the base memory address where a particular program begins storing its generated text or data in coherent form (i.e. the base address of its text buffer), that address can be inserted into the SAVExxxx.ASM file as the operative 'BEGTEXT EQU' statement to make the program specific for rescuing from memory text created by or brought into memory by that program.

Only one EQU statement for BEGTEXT (the beginning address of the text in memory) can be operative at a time for assembly of the source (.ASM) file with ASM.COM to work without error. In the .ASM file, insert a ";" or "*" ahead of each EQU statement which is to be disabled, leaving only one "BEGTEXT EQU nnnnH" unblocked (active). Then use ASM.COM to assemble the file, and LOAD.COM to then convert the resultant .HEX file to an executable .COM file.

I have already twice used the program to rescue a MEMORITE file: it is much simpler to use SAVEMEM14 than to go into ROM to find the text, calculate its size, move it down to 0100H, figure the number of pages, and SAVE them into a rescue file. This program makes of all of that unnecessary -- it

just records to disk until it finds an EOF (end-of-text) character, or reaches BDOS+5 if an EOF is not found.

If you RENAME the resultant SALVAGED file to a .MEM filetype, you will find that MEMORITE will actually load the saved file -- despite its giving the message "ERROR IN ACCESSING DISK" and failing to bring up a file description: but after hitting ESC twice following a MEMORITE R[ead] command, you will find that the file has indeed been read into memory. Give it a new ID and description, and that will make it accessible to MEMORITE thereafter without the disconcerting error message.

The descriptive data of a .MEM file is not written to disk by the SAVMEMxx program(s) because that descriptor is not contained within the MEMORITE textbuffer area of memory, and so is not salvaged with the SAVMEMxx program. THE SAVMEMXX program would have to be substantially modified to enable it to pick up that descriptive material from the different area of memory where it is stored.

EPL FILES: An even more radical modification would be required to rewrite SAVExxxx to create a version which could save EXECUPLAN (and other spreadsheet) ASCII text -- because EPL files (at least) are stored in several parts, or tables, some of it 'backwards' or top-down, as a mirror image, with only part of its data being stored consecutively in memory. While the titles of EXECUPLAN v. 2 begin at about 5FF4H, and the table of primary addresses begins at 6185H, with other tables following that -- the ASCII text is stored in mirror-image (backwards) from the top of the text buffer [BDOS+5] downward instead of in the ascending order which SAVExxxx expects. I have not fully fathomed the memory storage scheme for EPL data, but from the above it becomes obvious that the SAVExxxx program would have to be substantially reworked for EXECUPLAN rescues.

SCOPE text is not always fully memory resident (nor is WordStar's). Therefore the program will save only whatever of the text is still in memory. The rest of the file will be found in \$\$\$ temporary file(s) on the disk, which you must also rescue if you would pull it all back together; otherwise a concatenation with the .BAK file may be necessary. After SCOPE writes out all of the memory text buffer, it initializes the buffer with the EOF character 1AH. If you use SAVSC19.COM on that, you will get an empty SALVAGED file -- for the SAVEXXXX program quits upon finding an EOF 1AH, or an 0FFH or a NULL 00.

I found the Weinberger program a neat, easy to use rescue utility -- a version of which (for each type of program you use, on which it can be used) should be on every program disk and utility disk. His article is a useful tutorial on assembly language. It might be too much if you have never before stuck a toe into the waters of assembly language: but even so, you could perhaps get something out of it by just typing the .ASM source code into a file and then assembling and loading

it (as he instructs) in order to produce a .COM file. But be prepared for some frustration if you have not typed the .ASM file in exactly as written [inserting every comma, noting and respecting the absence of spaces, the presence of semi-colons, etc.] and you cannot spot the error(s) or omission(s). Deciphering the Assembly error messages and trying to locate the error lines is not always fun! Assembly - time errors are often anything but obvious.

Editor's note - Ed's work is most welcome, especially to those 57% of you who are using the MEMORITE word-processor. I've tried the MEMINDX programs and they work fine. Maybe we can talk Ed into parting with the latest versions - and source code. All of Ed's files are on CP/M library disk 1918.

□ □ □

STATUS OF SYSTEM/Z

by Buzz Rudow

Several people have written and called to ask about Bob Zale and the status of System/z. I've tried many phone calls. Even sent a registered letter. It was picked up, but never answered.

Two weeks ago I tried again. The answering service is still there. I left the message "Would you please have anyone in the office, not just Bob Zale, call me collect, at any time, and inform me of the current status of System/z product availability". No response. If any of you can add to my knowledge, please call or write.

□ □ □

CLASSIFIED

The Classified Section of the MUG Newsletter is available to all users, free of charge. Just call or write us with a description of what hardware, software, information, etc., that you want to obtain, or dispose of.

FOR SALE - Micropolis 1263 8" rigid disk. Forty-five megabytes for \$1195, plus shipping.

Lynn or Buzz Rudow, 604 Springwood Cir., Huntsville AL 35803. Phone (205) 881-1697.

.....

WANTED - Documentation for: (1) Vector Graphic Extended Systems Monitor, Version 3, Option EV-11 Copy in 2708 Eprom; (2) Vector Graphic Megastor Interface, Realtime Clock, Tape Back-up Interface, Part #3009; (3) Vector Graphic Floppy Disk/Hard Disk Controller, Part #3049; (4) Ithaca Intersystem Inc., Ia-2030, Rev. B, 64K, Dynamic Ram S-100 board.

David Beard, Route 2, Box 317-C, Fairhope AL 36532.

.....

MICROPOLIS/VG USERS GROUP Newsletter #74 September 1986

CLASSIFIED (continued)

FOR SALE - New Fulcrum Omnidisk controller with CP/M, \$195. The following are offered as is: Hayes S-100 Micromodem \$45; Vector Graphic 48K RAM and Z80 boards, \$20 each.

Gary Van Cott, P.O. Box 1879, Grafton VA 23692.
Phone (804) 898-3680.

WANTED - S-100 2-serial RS-232 and 2-parallel board, with specifications and documentation.

C. T. Frost, P.O. Box 668, Buda TX 78610. Phone (512) 295-5610.

FOR SALE - Three Vector Graphic MZ 64K CP/M Computer Systems with three Mindless Terminals. \$750 for all, or best offer.

Mike Simon, (212) 688-9691.

FOR SALE - One external drive system for a Vector Graphics computer. Bought at auction. Has one

floppy and one hard disk, model number DDW 5717. No documentation or interface cable. \$200.

Kelly Hancock, 5622 N. Atlantic, Portland, Oregon 97217. Phone (503) 285-8876.

FOR SALE - Vector Graphic 2600 System. Make offer.

David Lacey, (319) 353-2585.

MICROPOLIS/VECTOR GRAPHIC USERS GROUP NEWSLETTER

Published monthly by DAMAN, 604 Springwood Circle, Huntsville AL 35803-1740. Subscription rates: North America; \$18/year. Elsewhere; \$25/year, airmailed.

For assistance with general information, MUG/Vector Graphic, Micropolis drives & parts, call Lynn at (205) 881-1697 during the Central Times of 9AM to 5PM. Buzz is sometimes around in the evening (Lynn will know), and from 9-12 on Saturday.

MICROPOLIS/VECTOR GRAPHIC
USERS GROUP
A Division of **DAMAN**
604 Springwood Circle
Huntsville AL 35803-1740
(205) 881-1697



FIRST CLASS MAIL

FIRST CLASS MAIL